Requested Patent: JP2002111674A

Title:

METHOD AND SYSTEM FOR MANAGING CONNECTION AND COMPUTER READABLE RECORDING MEDIUM WITH PROGRAM FOR REALIZING THAT METHOD RECORDED THEREON ;

Abstracted Patent: JP2002111674 ;

Publication Date: 2002-04-12 ;

Inventor(s): GOTO YASUNOBU ;

Applicant(s): HITACHI LTD ;

Application Number: JP20000297071 20000926 ;

Priority Number(s): ;

IPC Classification: H04L12/28; G06F13/00; H04L12/56 ;

Equivalents: ;

ABSTRACT:

PROBLEM TO BE SOLVED: To make possible to identify a normally communicable connection between server processes when an extra server machine restarts after shutdown in a client server system performing online transaction where the function of the server is distributed to a plurality of server machines.SOLUTION: When a server process establishes connection with the server process of an extra server machine, the time acquired from a system is registered as a connection establishment time in a table 11 managing connection. A normally communicable connection can thereby be identified by checking the connection establishment time even when the extra server machine restarts after shutdown.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2002－111674
（P2002－111674A）
(43)公開日　平成14年４月12日(2002.4.12)

(51)Int.Cl.⁷	識別記号	ＦＩ		テーマコード（参考）
Ｈ０４Ｌ　12/28		Ｇ０６Ｆ　13/00	３５３Ｃ	５Ｂ０８９
Ｇ０６Ｆ　13/00	３５３	Ｈ０４Ｌ　11/00	３１０Ｄ	５Ｋ０３０
Ｈ０４Ｌ　12/56		11/20	１０２Ａ	５Ｋ０３３

審査請求　未請求　請求項の数５　ＯＬ　（全 12 頁）

(21)出願番号　特願2000－297071(P2000－297071)

(22)出願日　平成12年９月26日(2000.9.26)

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　後藤　康信
神奈川県小田原市国府津2880番地　株式会社日立製作所ストレージシステム事業部内
(74)代理人　100075096
弁理士　作田　康夫

Ｆターム(参考)　5B089 GA11 GA21 GB02 JA13 KA12 KG03 KG04
5K030 HB16 HB19 HC01 HC14 JT02 KA02 LE02
5K033 BA04 DA13 DB12 DB14

(54)【発明の名称】　コネクション管理方法およびシステムおよび前記方法を実現するプログラムを記録したコンピュータ読み取り可能な記録媒体

(57)【要約】
【課題】サーバの機能（サービス）を複数のサーバマシンに分散したオンライントランザクション処理を行うクライアント・サーバシステムにおいて、別サーバマシンがダウンした後、再起動した場合に、正常に通信できるサーバプロセス間のコネクションを識別できることを目的とする。
【解決手段】サーバプロセスが別サーバマシンのサーバプロセスとコネクションを確立したとき、システムから取得した時刻をコネクション確立時刻として、コネクションを管理するテーブル１１に登録する。これにより、別サーバマシンがダウンした後、再起動した場合でも、コネクション確立時刻をチェックすることで、正常に通信できるコネクションを識別できる。

図４

１１　コネクション管理テーブル

ソケットディスクリプタ	接続先IPアドレス	接続先ポート番号	サーバプロセス名称	コネクション確立時刻 <秒>	<μ秒>
5	172.17.150.15	１１０００	Ｐ２０	965250920 (2000/08/03 09:02:00)	640910
6	172.17.150.32	１２０００	Ｐ３０	965251110 (2000/08/03 09:05:10)	638940

【特許請求の範囲】
【請求項１】コネクション管理方法において、第一のコネクション確立相手の計算機に関する情報および第一のコネクションに関する第一の識別子と、第二のコネクション確立相手の計算機に関する情報および第二のコネクションに関する第二の識別子とを格納し、前記第一の識別子と前記第二の識別子とを比較し、使用するコネクションを決定することを特徴とするコネクション管理方法。
【請求項２】コネクション管理方法において、第一のコネクション確立相手のプログラムに関する情報および第一のコネクションに関する第一の識別子と、第二のコネクション確立相手のプログラムに関する情報および第二のコネクションに関する第二の識別子とを格納し、前記第一の識別子と前記第二の識別子とを比較し、使用するコネクションを決定することを特徴とするコネクション管理方法。
【請求項３】請求項１記載のコネクション管理方法において、前記識別子とは計算機から取得した時刻、もしくはコネクションの計数値であることを特徴とするコネクション管理方法。
【請求項４】コネクション管理システムにおいて、第一のコネクション確立相手の計算機に関する情報および第一のコネクションに関する第一の識別子と、第二のコネクション確立相手の計算機に関する情報および第二のコネクションに関する第二の識別子とを格納し、前記第一の識別子と前記第二の識別子とを比較し、使用するコネクションを決定するプロセスを保持することを特徴とするコネクション管理システム。
【請求項５】請求項１記載の方法を実現するプログラムを格納したコンピュータ読み取り可能な記録媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、サーバの機能（サービス）を複数のコンピュータ（サーバマシン）に分散したクライアント・サーバシステムにおいて、再度確立したコネクションを識別して、通信を正常に行う方法に関するものである。
【０００２】
【従来の技術】サーバの機能（サービス）を複数のコンピュータ（サーバマシン）に分散し、ネットワークで接続したオンライントランザクション処理を行うクライアント・サーバシステムにおいて、各サーバマシンで動作するサーバプロセスは、プロセス起動時、別サーバマシンのサーバプロセスとの間に、ＴＣＰソケットを使用してコネクションを確立することで、自サーバプロセスの機能（サービス）が実行可能であることを別サーバマシンのサーバプロセスに通知する。
【０００３】各サーバプロセスは、プロセスのローカルメモリ上にコネクション管理テーブルを持ち、コネクション確立が完了したとき、コネクション管理テーブルにコネクション情報を登録する。コネクション管理テーブルに登録するコネクション情報としては、確立したコネクションのＴＣＰソケットのソケットディスクリプタ、接続先ＩＰアドレス、接続先ポート番号、サーバプロセス名称である。
【０００４】サーバプロセスは、サーバプロセス間で確立するコネクションを１つとし、また、オンライントランザクション処理を行うサーバプロセスには、性能と信頼性が要求されることから、確立したコネクションは永続的に保持する。コネクション管理テーブルにコネクション情報が登録されていることで、別サーバマシンの該当サービスの呼び出しが可能であることを意味する。
【０００５】オンライントランザクション処理を行うクライアント・サーバシステムでは、クライアントからサービス要求（問い合わせ電文送信）が行われると、該当するサービスのサーバプロセスは、要求されたサービスの処理を行い、その結果を応答電文としてクライアントに送信する。
【０００６】クライアントからのサービス要求内容によっては、サーバの処理が複数サーバマシンのサーバプロセスの機能によって実現される場合がある。この場合、クライアントから直接サービス要求されたサーバプロセスは、サーバプロセス起動時に確立したコネクションを使用して、別サーバマシンのサーバプロセスとの間で、サービス要求（問い合わせ電文の送信、応答電文の受信）を行う。
【０００７】オンライントランザクション処理のクライアント・サーバシステムでは、サーバプロセスは、複数のクライアントからの要求を処理しなければならないため、特定のサービス処理でブロックしてはならない。
【０００８】この対処方法として、ＴＣＰソケットを使う通信部分では、ＴＣＰソケットを非ブロッキングモードに設定する。
【０００９】非ブロッキングモードのＴＣＰソケットを使用することにより、データ送信処理（例えば、接続したソケットへデータを送信するｓｅｎｄ（）ソケット関数の発行）を行った場合、オペレーティングシステム（カーネル）が管理している送信用ソケットバッファに空きがあると、送信用ソケットバッファに送信データがコピーされた時点で、カーネルから制御が戻り、通信処理でブロックすることなく処理を続行できる。
【００１０】従来、これらのＴＣＰソケットを使用するプロセス間通信については、「ＵＮＩＸネットワークプログラミング第2版　Ｖｏｌ．１」（Ｗ．リチャード・スティーブンス著、篠田陽一訳、トッパン、１９９９年発行）において論じられていた。
【００１１】
【発明が解決しようとする課題】ＴＣＰソケットを使用したプロセス間通信では、通信を行っていない状態で相

手サーバマシンがクラッシュ（サーバマシンがダウン）した場合（マシンがダウンした場合）、コネクション切断を示すデータ（パケット）が送信されてこないため、確立済みコネクションには何も通知されない（つまり、コネクション管理をするプロセス若しくはプログラムに通知がないため、自サーバマシンでコネクションが使用できるか否かが確認できない）。

【００１２】その後，相手サーバマシンが再起動されて、そのサーバマシンでサーバプロセスが再起動されると、サービスが実行可能であることを通知するため、再度、そのサーバプロセスからコネクション確立要求が行われて，新しくコネクションが確立される。

【００１３】すると、相手サーバマシンのクラッシュを知らないまま、再起動後のプロセスとコネクション確立を行ったサーバプロセスのコネクション管理テーブルには、再起動前に確立したコネクション情報（接続先ＩＰアドレス、接続先ポート番号、サーバプロセス名称）が登録されたまま、再起動後に確立したコネクション情報（接続先ＩＰアドレス、接続先ポート番号、サーバプロセス名称）が登録されることになり、同じサーバプロセスに関するコネクション情報が重複してしまい、再起動前の通信不可能なコネクションと再起動後に確立した正常なコネクションを識別できないという問題がある。

【００１４】もし、接続先プロセス（ソケット）が存在しなくなった再起動前のコネクションに、データ送信を行った場合、非ブロッキングモードのＴＣＰソケットでは、データ送信のソケット関数が正常リターンし、データ送信が正常にできたように見えるが、実際には送信用ソケットバッファに送信データを書き込んだだけであり、送信相手のＴＣＰまたはプロセスにデータを送れたということではない。

【００１５】したがって、相手サーバマシンのクラッシュを知らないサーバプロセスで、再起動前の通信不可能なコネクションに問い合わせ電文送信を行った場合、正常に電文送信ができたように見えるので、応答電文が送信されてくるのを監視する。

【００１６】しかし、再起動前の通信不可能なコネクションに送信した問い合わせ電文は再起動後のサーバプロセスに届くことはなく、応答電文も送信されてこないため、問い合わせ電文送信を行ったサーバプロセスでは、サービス要求のタイムアウトが発生する。つまり、再起動後のサーバプロセスが正常に動作している状態であるのに、問い合わせ電文送信を行ったサーバプロセスでは、再起動後のサーバプロセスとの通信エラー（サービス要求のタイムアウト）が発生したようになる。

【００１７】また、「ＵＮＩＸネットワークプログラミング第２版　Ｖｏｌ．１」（Ｗ．リチャード・スティーブンス著、篠田陽一訳、トッパン、１９９９年発行）には、通信を行っていないときの相手サーバマシンのクラッシュを検出する手段として、ＴＣＰソケットのＳＯ＿ＫＥＥＰＡＬＩＶＥオプションが論じられている。しかし、ＳＯ＿ＫＥＥＰＡＬＩＶＥオプションは、パケットの送受信が２時間行われなかった場合、ＴＣＰが生存確認用パケットを送信し、さらに１１分１５秒経過しても、その応答パケットが返ってこなかったとき、相手サーバマシンがクラッシュ（サーバマシンがダウン）したものと判断し、上位アプリケーションに通知するものである。

【００１８】したがって、ＴＣＰソケットのＳＯ＿ＫＥＥＰＡＬＩＶＥオプションを使用しても、相手サーバマシンがクラッシュ（サーバマシンがダウン）したことを即時に認識できるわけではないため、オンライントランザクション処理のクライアント・サーバシステムのサーバプロセス間通信には向いていない機能であり、上記問題を解決できない。

【００１９】以上のように、従来技術は、別サーバマシンがクラッシュ（別サーバマシンがダウン）して、その後サーバプロセスが再起動された場合についての記載がなかった。

【００２０】本発明の目的は、正常に通信できるコネクションを識別することにある。

【００２１】

【課題を解決するための手段】本発明のコネクション管理方法において、第一のコネクション確立相手の計算機又はプログラムに関する情報および第一のコネクションに関する第一の識別子と、第二のコネクション確立相手の計算機又はプログラムに関する情報および第二のコネクションに関する第二の識別子とを格納し、前記第一の識別子と前記第二の識別子とを比較し、使用するコネクションを決定する。

【００２２】尚、本発明のコネクション管理方法において、前記識別子とは計算機から取得した時刻、もしくはコネクションの計数値などコネクションを識別し、決定するための何らかの情報である。

【００２３】又は、本発明のコネクション管理方法において、第一のコネクション確立相手の計算機に関する情報および第一のコネクションに関する第一の識別子と、第二のコネクション確立相手の計算機に関する情報および第二のコネクションに関する第二の識別子とを格納し、前記第一の識別子と前記第二の識別子とを比較し、使用するコネクションを決定するプロセスを保持する。

【００２４】さらに、上記目的を達成するために、各サーバプロセスが持っているコネクション管理テーブルに、コネクション確立時刻を追加し、コネクション確立要求を受け付ける処理で、コネクション確立後、システム時刻を取得し、ソケットディスクリプタ、接続先ＩＰアドレス、接続先ポート番号、プロセス名称、および取得したシステム時刻をコネクション管理テーブルに設定し、時刻をチェックすることで、正常に通信できるコネクションを識別できるようにしたものである。

【００２５】また、本発明を実現するプログラムを、コンピュータ読み取り可能な記録媒体（ハードディスク、各種メモリ、フロッピー（登録商標）ディスク、光磁気ディスク、ＣＤ－ＲＯＭ、磁気テープなどの記録媒体）に格納し、その媒体から本発明を実現するプログラムを読み出して本発明を実施しても良いし、前記記録媒体に格納したプログラムにネットワーク等を通じてアクセスして本発明を実施しても良い。また、ネットワークを通じて本発明を実現するプログラムを格納した計算機（もしくは記録媒体）から本発明を実現するプログラムをダウンロードして、別な計算機上の計算機（もしくは別な記録媒体）に本発明を実現するプログラムをインストールして本発明を実施しても良い。

【００２６】

【発明の実施の形態】以下、本発明の実施例について、図を使用して説明する。

【００２７】図１は、本発明によるサーバの機能（サービス）を複数のコンピュータ（サーバマシン）に分散し、ネットワークで接続したオンライントランザクション処理を行うクライアント・サーバシステムの構成図である。図１において、クライアント４とサーバマシン１、サーバマシン２、サーバマシン３があり、これらはネットワーク８によって接続されている。

【００２８】ネットワーク８は、ＬＡＮ（Ｌｏｃａｌ Ａｒｅａ Ｎｅｔｗｏｒｋ）であるが、ＷＡＮ（Ｗｉｄｅ Ａｒｅａ Ｎｅｔｗｏｒｋ）とすることもできる。また、ネットワーク８に接続されるクライアント４、サーバマシン１、２、３との通信には、ＴＣＰ／ＩＰ（Ｔｒａｎｓｍｉｓｓｉｏｎ Ｃｏｎｔｒｏｌ Ｐｒｏｔｏｃｏｌ／Ｉｎｔｒｎｅｔ Ｐｒｏｔｏｃｏｌ）が使用される。

【００２９】クライアント４では、トランザクション要求を行うアプリケーションソフトウェアが動作し、必要となるサービス要求に応じて、サーバマシンのサーバプロセスへＴＣＰソケットを使用して、サービス要求（問い合わせ電文の送信および応答電文の受信）が行われる。

【００３０】サーバマシン１、２、３では、それぞれサービスの機能が異なるサーバプロセス１０、サーバプロセス２０、サーバプロセス３０が動作する。サーバプロセス１０、２０、３０には、それぞれ別サーバマシン構成定義ファイル５、６、７が用意されている。

【００３１】別サーバマシン構成定義ファイル５、６、７は、当該サーバマシンのサーバプロセスが動作可能であることを通知すべき別サーバマシン上のサーバプロセスの情報（別サーバプロセス定義テーブル５１、６１，７１）がそれぞれ格納されている。

【００３２】図２は、別サーバマシン構成定義ファイルに格納される別サーバプロセス定義テーブルの一例を示す図である。別サーバプロセス定義テーブル５１は、当該サーバマシンのサーバプロセスが起動時にコネクション確立する別サーバマシン上のサーバプロセスのプロセス名称、ポート番号、ＩＰアドレスから構成される。

【００３３】プロセス名称は、別サーバマシンで動作するサーバプロセスの名称である。ポート番号は、別サーバマシンで動作するサーバプロセスの受付用ポート番号である。ＩＰアドレスは、そのサーバプロセスが動作するサーバマシンのＩＰアドレスである。

【００３４】図３は、各サーバマシンで動作するサーバプロセスの構成図である。サーバプロセス１０は、サービス本体部１２、通信制御部１３から構成され、通信制御部はコネクション管理テーブル１１を持つ。

【００３５】次にこれらの要素について詳細に説明する。サービス本体部１２は、要求されたサービスを処理する本体部分であり、処理の中核部分については各サーバマシンのサーバプロセスごとで異なるが、サーバプロセスの構成上は同じであり、サービス処理１００から構成される。通信制御部１３は、ＴＣＰソケットを使用して、クライアントおよび別サーバマシンで動作するサーバプロセスとの送受信を行う部分であり、通信準備処理２００、電文送信処理３００、事象発生待ち処理４００から構成され、コネクション管理テーブル１１を使用して処理を行う。

【００３６】図４は、サーバプロセスがローカルメモリに確保するコネクション管理テーブルの一例を示す図である。

【００３７】コネクション管理テーブル１１は、別サーバマシンのサーバプロセスとの間で、コネクション確立に使用したＴＣＰソケットのソケットディスクリプタ、接続先ＩＰアドレス、接続先ポート番号、サーバプロセス名称、コネクション確立時刻から構成される。

【００３８】ソケットディスクリプタは、ＴＣＰソケットを使用してコネクションを確立したときに、オペレーティングシステムによって決定された識別子である。

【００３９】接続先ＩＰアドレスは、ＴＣＰソケットを使用して確立したコネクションの接続先プロセスが動作しているサーバマシンのＩＰアドレスである。コネクション確立要求を行った側では、コネクション確立要求時に求めた情報であり、コネクション確立要求を受けた側では、受付用ソケットのキューからコネクション確立要求を取り出しコネクションを確立するａｃｃｅｐｔ（）ソケット関数で取得した情報である。

【００４０】接続先ポート番号は、ＴＣＰソケットを使用して確立したコネクションの接続先プロセスの受付用ポート番号である。コネクション確立要求を行った側では、コネクション確立要求時に求めた情報であり、コネクション確立要求を受けた側では、コネクション確立要求元から最初に送信されるポート番号電文に設定された受付用ポート番号である。

【００４１】サーバプロセス名称は、ＴＣＰソケットを

使用して確立したコネクションで接続しているプロセスのプロセス名称であり、接続先ＩＰアドレス、接続先ポート番号から求めた情報である。

【００４２】コネクション確立時刻は、コネクション確立時に取得したシステム時刻（例えば、１９７０年1月1日、００：００：００を基準とし、この時刻から起算した秒数とマイクロ秒数）である。

【００４３】図5は、図3のサービス本体部１２における、サービス処理の流れを示す図である。

【００４４】サービス処理１００では、プロセスが起動されると、サービス本体の準備を行い（ステップ１０１）、通信準備処理を行って（ステップ１０２）、事象発生待ち処理でクライアントまたは別サーバマシンのサーバプロセスからのサービス要求事象の発生を待つ（ステップ１０３）。サービス要求事象が発生すると、サービス実行処理でサービスを実行して応答電文を作成し（ステップ１０４）、応答電文の電文送信処理を行い（ステップ１０５）、事象発生待ち（ステップ１０３）の状態に戻る。

【００４５】図6は、図5におけるサービス処理の中（ステップ１０２）から呼び出される、通信準備処理の流れを示す図である。

【００４６】通信準備処理２００では、動作しているサーバマシン名称を取得するｇｅｔｈｏｓｔｎａｍｅ（）関数を発行して、サーバマシン名称を取得し（ステップ２０１）、指定したサーバマシン名称に関する情報を取得するｇｅｔｈｏｓｔｂｙｎａｍｅ（）関数を発行して、動作しているサーバマシンのＩＰアドレスを取得し（ステップ２０２）、指定したサービス名称に関する情報を取得するｇｅｔｓｅｒｖｂｙｎａｍｅ（）関数を発行して、自サーバプロセスの受付用ポート番号を取得し（ステップ２０３）、ＴＣＰソケット通信で必要なソケットを生成するｓｏｃｋｅｔ（）ソケット関数を発行して、ソケットディスクリプタを取得し（ステップ２０４）、ステップ２０２で取得したＩＰアドレス、ステップ２０３で取得した受付用ポート番号を指定して、ソケットにアドレスを割り付けるｂｉｎｄ（）ソケット関数を発行して、受付用ソケットにアドレス割り付けを行い（ステップ２０５）、他からのコネクション確立要求の受け入れ準備を行うｌｉｓｔｅｎ（）ソケット関数を発行して、クライアントおよび別サーバプロセスからのコネクション確立要求を受付可能状態にする（ステップ２０６）。次に、ローカルメモリ上にコネクション管理テーブルの領域を確保する（ステップ２０７）。そして、ステップ２０３で取得した受付用ポート番号を通知するポート番号電文を作成し（ステップ２０８）、別サーバマシン構成定義ファイルを読み込み（ステップ２０９）、別サーバマシンのＩＰアドレス、受付用ポート番号を求めて、別サーバマシン構成定義ファイルに格納されているプロセスに対し、当該サーバプロセスが動作可能であることを通知するため、ポート番号電文の電文送信処理を行い（ステップ２１０）、通信準備処理を終了する（ステップ２１１）。

【００４７】図7は、図6における通信準備処理の中（ステップ２１０）および図5のサービス処理の中（ステップ１０５）から呼び出される、電文送信処理の流れを示す図である。

【００４８】電文送信処理３００では、電文送信先ＩＰアドレス、送信先ポート番号と一致するコネクション管理テーブルのエントリをサーチする（ステップ３０１）。サーチした結果、コネクション管理テーブルに登録されていない場合（ステップ３０２）、ＴＣＰソケット通信で必要なソケットを生成するｓｏｃｋｅｔ（）ソケット関数を発行してソケットディスクリプタを取得後、指定したアドレスへコネクション確立を要求するｃｏｎｎｅｃｔ（）ソケット関数を発行して、電文送信先へコネクション確立要求を行う（ステップ３０３）。

【００４９】コネクション確立が成功し（ステップ３０４）、電文送信先が別サーバマシン構成定義ファイルに格納されているサーバプロセスの場合（ステップ３０５）、当該コネクションはコネクション管理テーブルに登録して管理すべきコネクションであるので、システム時刻（１９７０年1月1日　００：００：００から起算した秒数とマイクロ秒数）を取得し（ステップ３０６）、コネクション情報をコネクション管理テーブルに登録し（ステップ３０７）、電文を送信して（ステップ３０８）、処理を終了する（ステップ３０９）。

【００５０】電文送信先が別サーバマシン構成定義ファイルに格納されているサーバプロセスでない場合（ステップ３０５）、当該コネクションは、コネクション管理テーブルに登録して管理すべきコネクションではないので、電文を送信して（ステップ３０８）、処理を終了する（ステップ３０９）。

【００５１】コネクション確立が失敗した場合（ステップ３０４）、ソケットをクローズして（ステップ３１０）、処理を終了する（ステップ３０９）。コネクション管理テーブルをサーチした結果、電文送信先が登録されている場合（ステップ３０２）、電文を送信して（ステップ３０８）、処理を終了する（ステップ３０９）。

【００５２】図8は、図5のサービス処理の中（ステップ１０３）から呼び出される、事象発生待ち処理の流れを示す図である。

【００５３】事象発生待ち処理４００では、受付用ソケットおよび確立済みコネクションのソケットを指定して、指定した複数のソケットディスクリプタの内どれかに事象が発生するまで当該プロセスを待ち状態にするｓｅｌｅｃｔ（）関数を発行して事象が発生するのを待つ（ステップ４０１）。

【００５４】受付用ソケットにコネクション確立要求事象が発生した場合（ステップ４０２）、受付用ソケット

のキューからコネクション確立要求を取り出しコネクションを確立するａｃｃｅｐｔ（）ソケット関数を発行してコネクション確立を行い（ステップ４０３）、コネクション確立要求元の受付用ポート番号が設定されたポート番号電文を受信し（ステップ４０４）、ポート番号電文のポート番号が別サーバマシン構成定義ファイルに格納されている場合（ステップ４０５）、当該コネクションはコネクション管理テーブルに登録して管理すべきコネクションであるので、システム時刻（１９７０年1月1日　００：００：００から起算した秒数とマイクロ秒数）を取得し（ステップ４０６）、コネクション情報をコネクション管理テーブルに登録する（ステップ４０７）。

【００５５】その後、コネクション管理テーブルのサーチを行い、コネクション管理テーブルに登録したコネクション情報と接続先ＩＰアドレスおよび接続先ポート番号が一致し、かつ確立時刻が小さいエントリ（コネクション情報）を比較することで（ステップ４０８）、正常に通信できるコネクションを識別し、確立時刻が古いコネクションがあった場合はそのコネクションをクローズし（ステップ４０９）、ソケットへの事象発生待ち（ステップ４０１）の状態に戻る。

【００５６】コネクション管理テーブルに登録したコネクションと接続先ＩＰアドレスおよび接続先ポート番号が一致し、かつ確立時刻が小さいエントリ（コネクション情報）がなかった場合は（ステップ４０８）、ソケットへの事象発生待ち（ステップ４０１）の状態に戻る。

【００５７】ポート番号電文のポート番号が別サーバマシン構成定義ファイルに格納されていない場合（ステップ４０５）、当該コネクションはコネクション管理テーブルに登録して管理すべきコネクションではないので、電文受信を行って（ステップ４１０）、処理を終了する（ステップ４１１）。

【００５８】また、受付用ソケット以外のソケット（確立済みコネクションのソケット）に事象が発生した場合は（ステップ４０２）、電文受信を行って（ステップ４１０）、処理を終了する（ステップ４１１）。

【００５９】なお、上記の実施の形態においては、コネクション管理テーブルに登録したコネクション情報で、接続先ＩＰアドレスおよび接続先ポート番号が一致したエントリがあった場合、正常に通信できるコネクションを識別する情報として、コネクション確立時のシステム時刻を使用したが、コネクション管理テーブルにコネクション情報を登録する度にカウントアップするカウンタを用いてその計数値を使用してもよい。

【００６０】また、上記以外の識別子を用いてコネクションを識別し、使用すべきコネクションを決定しても良い。たとえば、使用可能なコネクションの識別子は「１」、使用不可能なコネクションの識別子は「０」というように、コネクションを確立した相手のコンピュータの情報に対し、コネクションを識別するための何らかの情報を付加して管理しても良い。なお、本発明においては、別サーバプロセス定義テーブル５１や、コネクション管理テーブルの情報１１等の情報は計算機に関する情報（ＩＰアドレス等）を示したが、プログラム（オブジェクト、プロセスなど）の固有の情報を用いてもよい。（例えば、通信相手となる別サーバプロセスのオブジェクトリファレンスと、その別サーバプロセスとのコネクションを識別する識別子とを対応付けて管理し、使用できるコネクションを決定しても良い。このように、コネクションを確立するプログラムの固有情報と確立したコネクションに対する識別子とを対応づけて管理することで、コネクション相手のものを識別する情報が、コネクション相手となる別サーバプロセスが起動している計算機に関連する情報以外の場合においても本発明を適用することができる。）さらに、本発明を実現するプログラムを、コンピュータ読み取り可能な記録媒体（ハードディスク、各種メモリ、フロッピーディスク、光磁気ディスク、ＣＤ－ＲＯＭ、磁気テープなどの記録媒体）に格納し、その媒体から本発明を実現するプログラムを読み出して本発明を実施しても良いし、前記記録媒体に格納したプログラムにネットワーク等を通じてアクセスして本発明を実施しても良い。また、ネットワークを通じて本発明を実現するプログラムを格納した計算機（もしくは記録媒体）から本発明を実現するプログラムをダウンロードして、別な計算機（もしくは別な記録媒体）に本発明を実現するプログラムをインストールして本発明を実施しても良い。

【００６１】本発明は、以上説明したように、別サーバマシンで動作するサーバプロセスからのコネクション確立要求が発生したとき、システムの時刻を取得することで、正常に通信できるコネクションを識別するようにした。したがって、別サーバマシンがクラッシュ（別サーバマシンがダウン）して、サーバプロセス再起動後にも発生していたサーバプロセス間の通信エラー（サービス要求のタイムアウト）を防止でき、通信不可能（障害）となっているコネクションを識別できる効果がある。

【００６２】

【発明の効果】本発明は、以上説明したように、コネクションを識別することにより、通信エラーを少なくできるという効果が得られる。

【図面の簡単な説明】

【図１】本発明によるサーバ機能を複数サーバマシンに分散したクライアント・サーバシステムの構成図である。

【図２】別サーバプロセス定義テーブルの一例を示す図である。

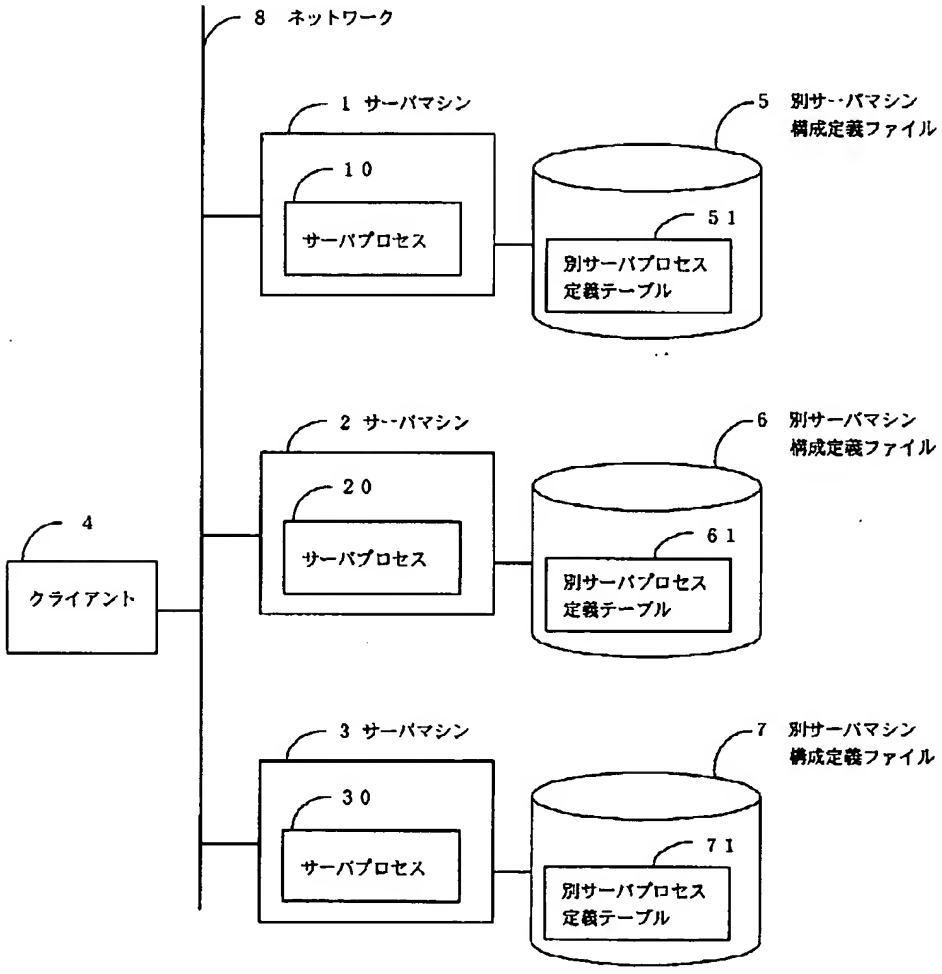
【図３】各サーバマシンで動作するサーバプロセスの構成図である。

【図４】コネクション管理テーブルの一例を示す図であ

る。
【図５】サービス処理を示す流れ図である。
【図６】通信準備処理を示す流れ図である。
【図７】電文送信処理を示す流れ図である。
【図８】事象発生待ち処理を示す流れ図である。
【符号の説明】
１０　サーバプロセス
１１　コネクション管理テーブル
１３　通信制御部
１００　サービス処理
２００　通信準備処理
３００　電文送信処理
４００　事象発生待ち処理

【図１】

図１



【図2】

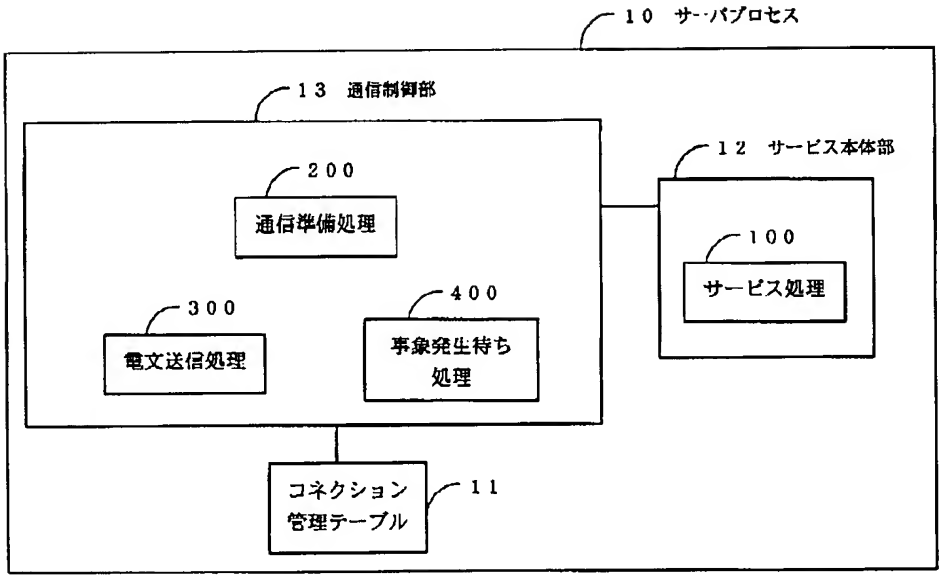
図2

51 別サーバプロセス定義テーブル

プロセス名称	ポート番号	ＩＰアドレス
ＳＰ２０	１１０００	172.17.150.15
ＳＰ３０	１２０００	172.17.150.32

【図3】

図3



【図4】

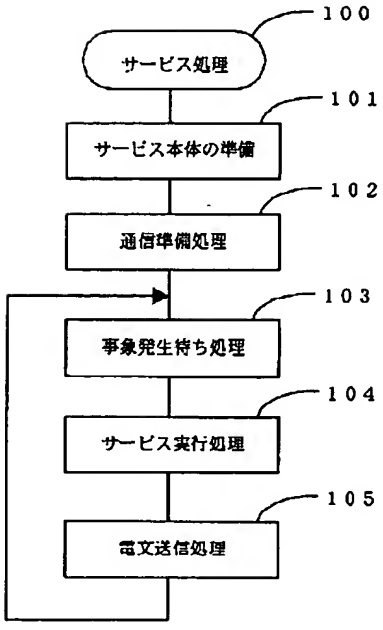
図4

11 コネクション管理テーブル

ソケット ディスクリプタ	接続先 IPアドレス	接続先 ポート番号	サーバ プロセス名称	コネクション確立時刻 <秒>	コネクション確立時刻 <μ秒>
5	172.17.150.15	11000	P20	965260920 (2000/08/03 09:02:00)	640916
6	172.17.150.32	12000	P30	965261110 (2000/08/03 09:05:10)	698946

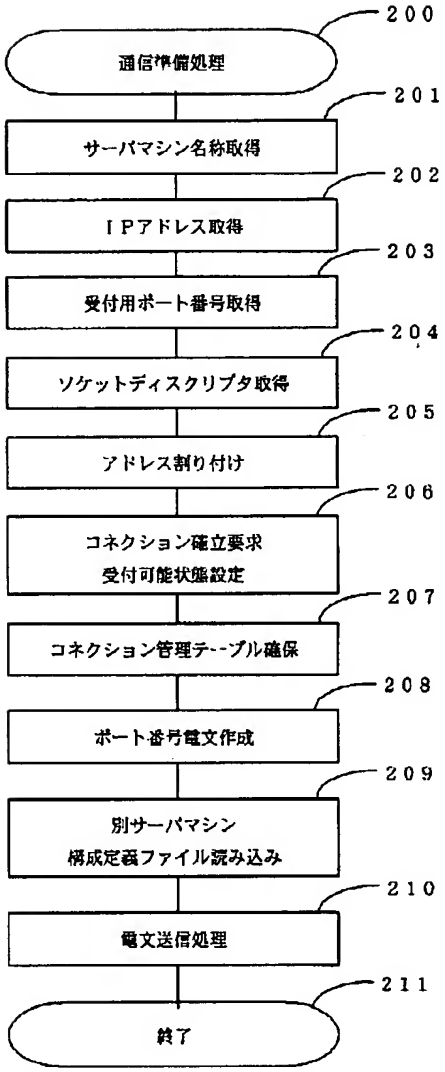
【図5】

図5



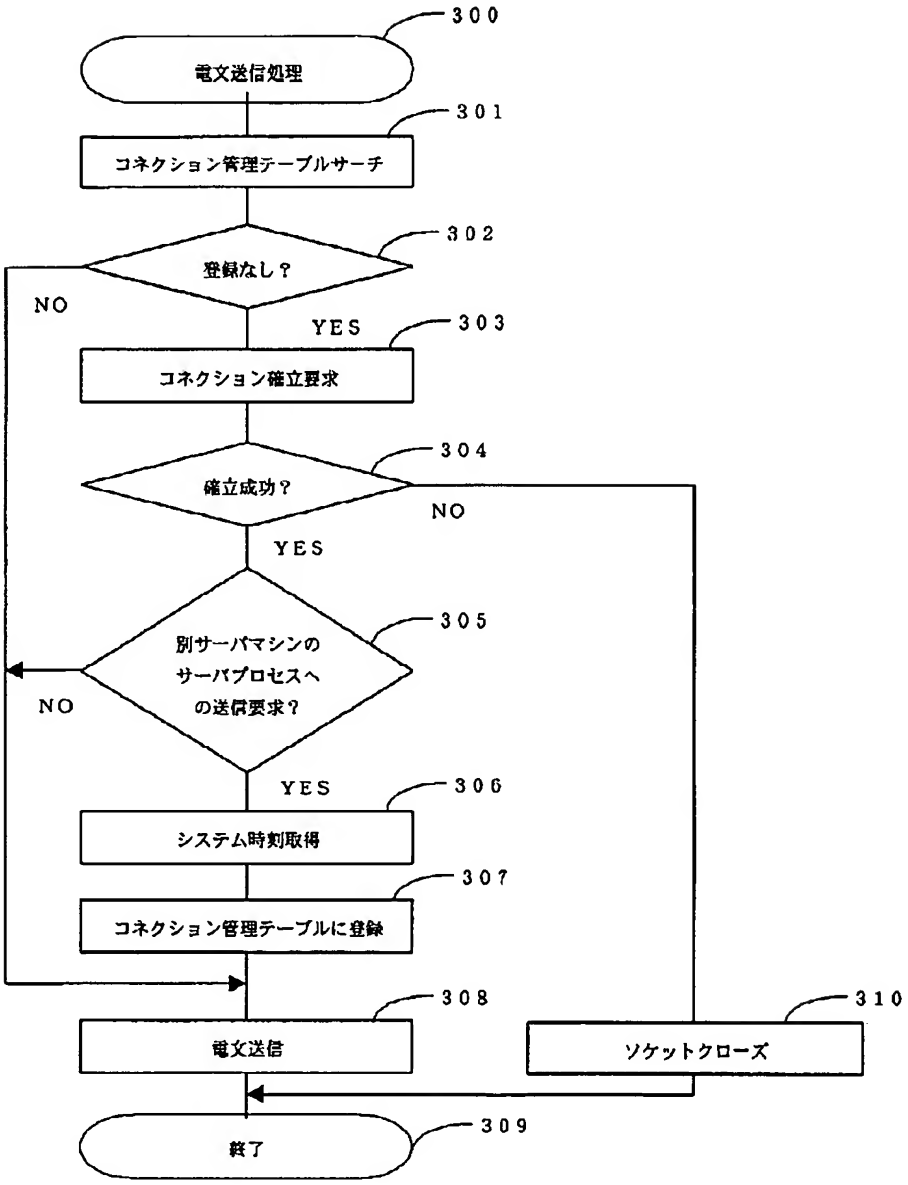
【図6】

図6


200
通信準備処理
201
サーバマシン名称取得
202
IPアドレス取得
203
受付用ポート番号取得
204
ソケットディスクリプタ取得
205
アドレス割り付け
206
コネクション確立要求
受付可能状態設定
207
コネクション管理テーブル確保
208
ポート番号電文作成
209
別サーバマシン
構成定義ファイル読み込み
210
電文送信処理
211
終了

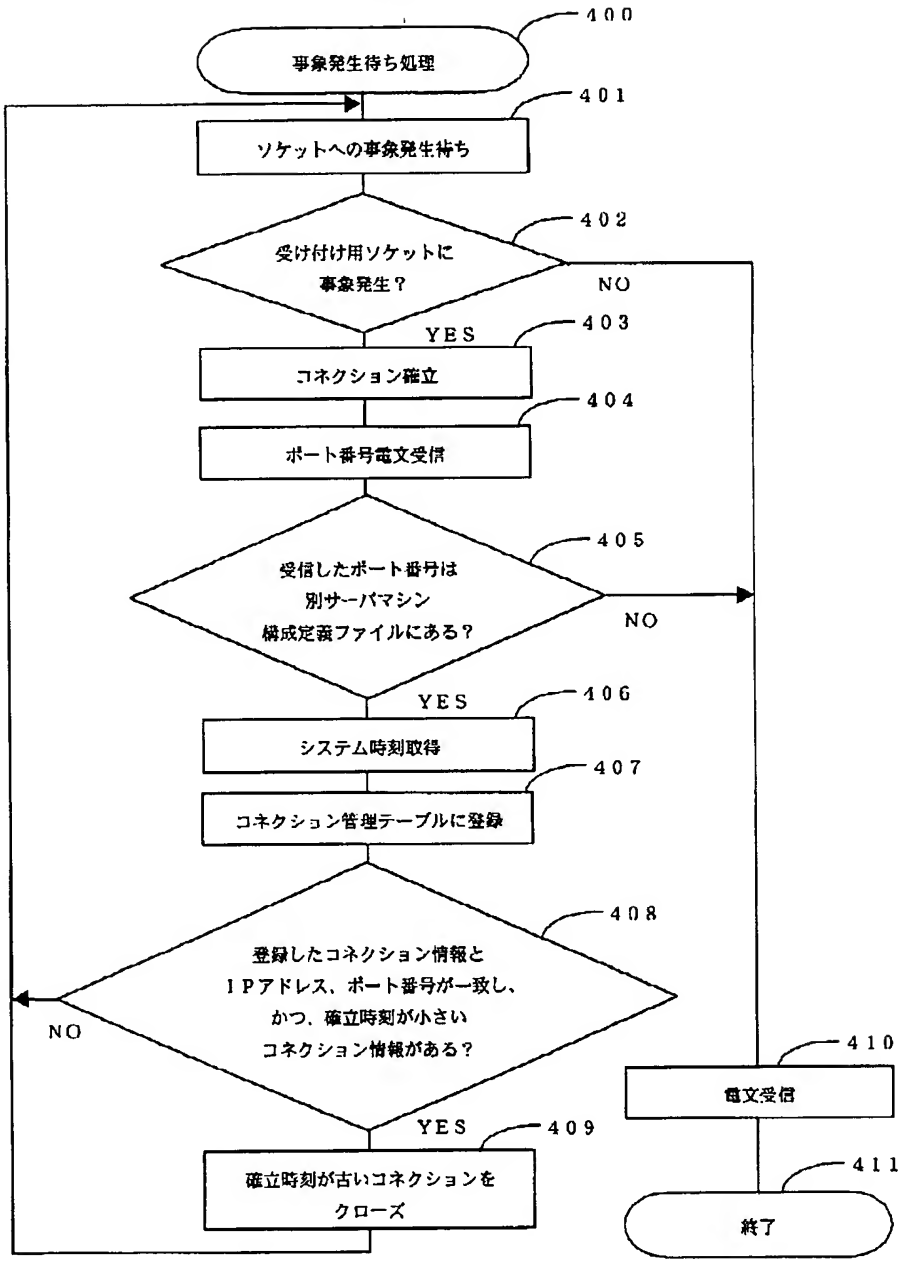
【図7】

図7


300
電文送信処理
301
コネクション管理テーブルサーチ
302
登録なし？
NO
YES
303
コネクション確立要求
304
確立成功？
NO
YES
305
別サーバマシンの
サーバプロセスへ
の送信要求？
NO
YES
306
システム時刻取得
307
コネクション管理テーブルに登録
308
電文送信
310
ソケットクローズ
309
終了

【図8】

図8


400
事象発生待ち処理
401
ソケットへの事象発生待ち
402
受け付け用ソケットに
事象発生？
NO
YES
403
コネクション確立
404
ポート番号電文受信
405
受信したポート番号は
別サーバマシン
構成定義ファイルにある？
NO
YES
406
システム時刻取得
407
コネクション管理テーブルに登録
408
登録したコネクション情報と
IPアドレス、ポート番号が一致し、
かつ、確立時刻が小さい
コネクション情報がある？
NO
YES
409
確立時刻が古いコネクションを
クローズ
410
電文受信
411
終了